Optimizing city-scale traffic through modeling observations of vehicle movements

Fan Yang, Alina Vereshchaka, Bruno Lepri, Wen Dong, *member, IEEE*

***Abstract*—The capability of traffic-information systems to sense the movement of millions of users and offer trip plans through mobile phones has enabled a new way of optimizing city traffic dynamics, turning transportation big data into insights and actions in a closed-loop and evaluating this approach in the real world. Existing research has applied dynamic Bayesian networks and deep neural networks to make traffic predictions from floating car data, utilized dynamic programming and simulation approaches to identify how people normally travel with dynamic traffic assignment for policy research, and introduced Markov decision processes and reinforcement learning to optimally control traffic signals. However, none of these works utilized floating car data to suggest departure times and route choices in order to optimize city traffic dynamics. In this paper, we present a study showing that floating car data can lead to lower average trip time, higher on-time arrival ratio, and higher Charypar-Nagel score compared with how people normally travel. The study is based on optimizing a partially observable discrete-time decision process and is evaluated in one synthesized scenario, one partly synthesized scenario, and three real-world scenarios. This study points to the potential of a "living lab" approach where we learn, predict, and optimize behaviors in the real world.**

I. Introduction

With 80% of newly sold vehicles in the U.S. able to communicate vehicle state through a telematics unit, and 57% of the population connected to the Internet by smart phones, datasets that track vehicles are increasingly available for transportation and policy researchers to study human mobility at scale. Such datasets contain rich information — from how drivers plan their daily activities and trips at the microscopic level to how road networks respond to transportation demand at the macroscopic level [1], [2], [3]. Traffic-information providers such as Google Traffic and INRIX can play an essential role in city traffic dynamics by suggesting optimal trip plans according to the observed movements of millions of vehicles. At the same time, artificial intelligence is accelerating workplace transition and the way people travel at a pace forecasting-based policy research might ultimately be unable to keep up with. This trend demands a paradigm that leverages traffic big data to deliver agile, quantifiable, and scalable solutions to our real-world transportation problems. Algorithms based on graphical models [4], [5] and neural networks [6], [7], [8], [9], [10] have been developed to make traffic predictions from the movement of millions of vehicles, but none of them utilizes these predictions to optimize complex city traffic dynamics. Similarly, optimization algorithms have been developed to identify how people normally travel through traffic assignment [11], [12], [13], [14] in policy research and to optimize traffic-light schedules [15], [16], [17], [18], [19], but none directly connects the observed vehicle trajectories with suggested driver departure times and route choices in a closed-loop control to optimize city traffic.

In this paper, we present a simulation study showing that by transforming the observed probe-vehicle movement data into traffic predictions and suggestions about optimal departure times and route choices in closed-loop control, we can achieve lower average trip time and higher on-time arrival ratio, and thus a higher Charypar-Nagel score [14], compared with how people normally travel. To achieve this, we model the traffic optimization problem as a partially observable Markov decision process, where the observations are the numbers of "probe" vehicles at road links and buildings, the future expected reward to optimize is the Charypar-Nagel scoring function [14], the control variables are related to departure times and route choices, and the dynamics are modeled as a queuing network approximated with a discrete-event model with neural network components. The simulation study is conducted using one synthesized scenario [14], one partly real and partly synthesized scenario [20], and three real-world scenarios [21], [22].

The uniqueness of this paper is that we combine machine learning methods and big floating car data to prototype a new traffic optimization approach. The variational tracking and optimal control algorithms that we developed [5], [23], [24] are complex and less straightforward, so we specifically develop a new particle filter algorithm to track and optimize traffic by extending our previous work [5], [25]. We also provide a detailed evaluation of this approach to traffic prediction and control in synthesized, partly real and partly synthesized, and real-world scenarios. Our approach not only simulates traffic jams during rush hours but also *predicts* from the trajectories of probe vehicles whether today's traffic jams will be formed earlier or last longer than usual, and helps drivers to *decide* and *plan* how to use the road network more efficiently.

A use-case diagram in Fig. 1 shows a high-level view of how the theory in this paper can be deployed in the real world to track traffic state and optimize traffic dynamics. Floating vehicle locations from drivers' navigation apps are aggregated

F. Yang was with the Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY, 14260 USA e-mail: (fyang24@buffalo.edu).

A. Vereshchaka was with the Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY, 14260 USA e-mail: (avereshc@buffalo.edu).

B. Lepri is with the Mobile and Social Computing Laboratory, Bruno Kessler Foundation, 38122 Trento, Italy (e-mail: lepri@fbk.eu).

W. Dong was with the Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY, 14260 USA e-mail: (wendong@buffalo.edu).

Manuscript received Month date, year; Month date, year

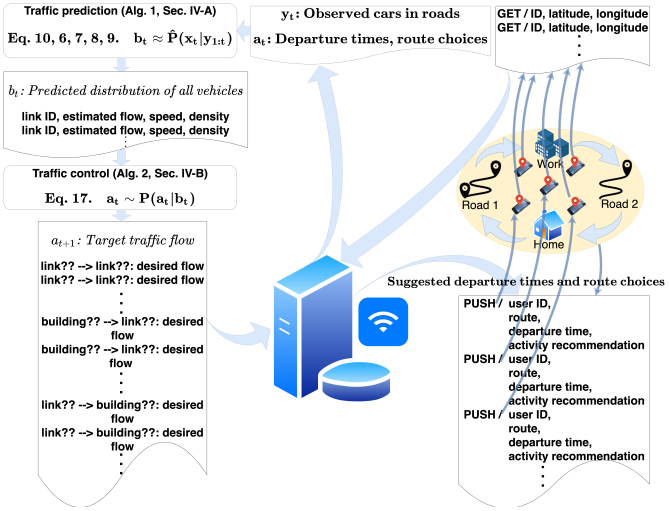


Fig. 1. A use-case diagram of traffic monitoring and control with drivers' navigation apps at high level.

into noisy observations about traffic demands and state (y_t and a_{t-1}). These noisy observations are fed into the traffic prediction algorithm (Algorithm 1, Sec. IV-A) to calculate the improved traffic state estimation as $b_t \approx \hat{P}(x_t|y_{1:t})$ using Eq. 10, which in turn uses Eqs. 6, 7, 8, and 9. The improvement comes from aggregating past observations $y_{1:t}$, and using a Bayesian formulation $\hat{P}(x_t|y_{1:t})$. The improved traffic state estimation is then fed into the traffic control algorithm (Algorithm 2, Sec. IV-B) to calculate the desired/target traffic flow as $a_t \sim P(a_t|b_t)$ using Eq. 17. The desired traffic flow control is in the form of what proportions of traffic should be directed to downstream links/facilities and is used to suggest departure times and route choices through drivers' navigation apps. This improves traffic with better and real-time information b_t about traffic state.

The remainder of this paper is organized as follows. Section II discusses other research efforts on making predictions and identifying optimal controls in the road transportation network from noisy observations. Section III introduces the discrete-event decision process to define the problem of optimizing city traffic dynamics. Section IV details a particle filter algorithm that predicts and optimizes traffic from noisy observations. Section V evaluates the performance of a discrete-event decision process and particle filter against other model-based and model-free methods on synthesized and real-world datasets. In Section VI we discuss the implications and limitations of our work and draw some conclusions.

II. Related Works

In this paper, we apply a discrete-event decision process [5], [23], [24] to predict complex city traffic dynamics from the trajectories of probe vehicles and accordingly optimize that traffic through departure times and route choices. Related research in intelligent transportation systems falls into two research streams: traffic prediction and traffic optimization.

Traditionally, traffic prediction was based on traffic cameras, inductive-loop traffic detectors, and similar technologies installed at fixed locations to capture speed, flow, and density. Algorithms include extended Kalman filter [26], localized extended Kalman filter [27], extended generalized Treiber-Helbing filter [28], and particle filter [29]. More recently, mobile phones and the Internet of Things have provided a new way to log the trajectories of millions of vehicles. These trajectories introduce an unprecedented opportunity to estimate home and work locations [2], identify trip purposes and special events [30], model driver behaviors [31], and track road network dynamics [3], [9]. Algorithms to leverage these trajectory data include probabilistic Bayesian networks [4], deep neural networks [6], [7], convolutional neural networks [8], [9], graph convolutional networks [32], [33], and recurrent neural networks [10]. Traffic prediction has been used for model-calibration in policy research, controlling traffic signals, and informing drivers. However, to the best of our knowledge the research in this paper is the first to optimize traffic by suggesting optimal departure times and route choices. It is also new to apply an agent-based [1] discrete-event model to both predict traffic and visualize how vehicles move in a city-scale road network in accordance with where probe vehicles move and how traffic policies change vehicle behavior.

Traffic optimization is conducted for transportation forecasting and traffic control. Transportation forecasting is a four-step process — trip generation, trip distribution, mode choice, and traffic assignment — that estimates the future usage of specific transportation facilities in order to assist policy research, such as assessing land-development impact [34]. Traffic optimization is conducted in the fourth steps through either mathematical programming [11], [12] or simulation [13], [35], [14], with the assumption that people select routes with the minimum travel times at the equilibrium. While optimization is used to identify reasonable routes in traffic assignment, its usage in this paper is to optimize the overall utilities of all people in a transportation network in response to instant traffic prediction based on the observed probe-vehicle locations. Our utility to optimize is the Charypar-Nagel scoring function [14], which entails minimizing not only travel time but also uncertainty in arrival time, as well as the impact of traffic fluctuation on planned activity duration. Various approaches have been applied to optimize traffic-signal control, including mixed-integer linear programming [15], model predictive control [16], Q-learning [17], policy gradient [18], and multi-agent reinforcement learning [19]. Numerous works have used Markov decision processes and reinforcement learning to optimize other aspects of transportation, such as the accessibility of taxicabs [36], [37]. While existing research optimizes transportation dynamics by setting traffic-light schedules, we optimize by advising drivers about ideal departure times and route choices.

As city-scale floating car data is becoming available to the academia [22], [38], we previously developed a variational solution to the intractable problem in complex-network optimal control [5], [23], [24]. The essence of the variational solution is to both find optimal trip plans in a tractable surrogate optimization problem and establish a mathematical guarantee that the performance of the solution in the original problem is not worse. In comparison with our previous works, the

focus in this paper is developing conceptually straightforward algorithms and providing comprehensive evaluation in the transportation domain.

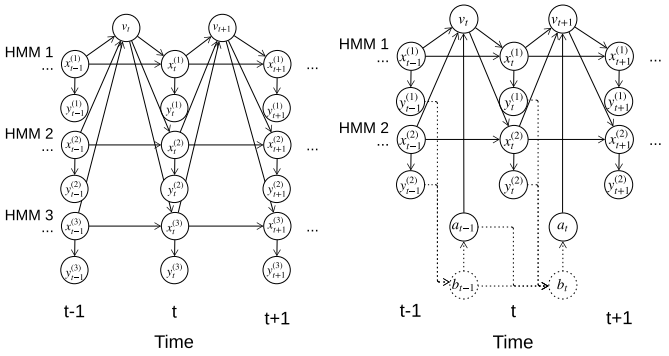
III. Problem Statement

In this Section, we introduce a discrete-event decision process (Eq. 2, Sec. III-A) to define the traffic optimization problem. In doing so, we specify the utility to optimize, the states, the events, the observations, and the control variables (Sec. III-B).

Overall, our purpose is to optimally control transportation dynamics in order to a) minimize the total travel time, b) minimize the delay for cars to arrive at their destinations due to traffic fluctuations, and c) minimize the effect of commuting on traffic congestion so that people can perform activities at destinations for an ideal amount of time without incurring significant travel-time increases. Optimal control is achieved by advising individual drivers about downstream links in route planning and the time to leave a given location. The resulting effect is stochastic. The performance indicators of a transportation network are calculated and estimated by the states of all the vehicles and probe vehicles, respectively, where the probe vehicles account for only a small fraction of the vehicle population. Optimal control is also computed from the observed probe-vehicle populations at different locations. This kind of control is applicable where a traffic-information provider (such as Google Maps or a traffic authority) provides trip plans according to the locations reported by a small number of drivers. We assume discrete-event dynamics, where the system state consists of numbers of vehicles at road links and buildings and is driven by elementary events in the form of an individual vehicle moving from one location to the next.

A. Discrete-Event Model for Inference and Decision Making

Here, we introduce a discrete-event model called the *stochastic kinetic model* [39], [40], which captures the dynamics of a complex social system driven by a set of events. A discrete-event model is a versatile model for describing a wide range of dynamics in various fields. It has many other names, including stochastic kinetic model [39], [40], stochastic Petri net [41], system dynamics model [42], multi-agent model specified through a flow chart or a state chart [43], Markov jump process [44], [45], continuous time Bayesian network [46], and production rule system [47]. The premise of introducing a discrete-event simulation model [48], [40], [43] to specify road network dynamics is that complex system dynamics can be described by a set of microscopic events that individually bring only minimal changes but in sequence induce complex behaviors. Using a discrete-event model, we specify traffic dynamics in a road network with a set of stochastic events — a driver starting a trip, moving to the next road, and ending a trip, for example — and we introduce a set of control variables to influence driver choices in response to the environment.



(a) Discrete event process (b) Discrete event decision process

Fig. 2. Graphical model representations of (2a) a discrete event process and a (2b) a discrete event decision process. A discrete event model captures complex system dynamics and specifies a decision-making problem compactly with a set of events.

Let $\mathbb{X}^{(1)}, \cdots, \mathbb{X}^{(M)}$ denote the individuals belonging to the M species in the system. A discrete-event process is generated by a set of events in the form of a production

$$\alpha_v^{(1)}\mathbb{X}^{(1)}+...+\alpha_v^{(M)}\mathbb{X}^{(M)} \xrightarrow{c_v} \beta_v^{(1)}\mathbb{X}^{(1)}+...+\beta_v^{(M)}\mathbb{X}^{(M)}. \quad (1)$$

This production is interpreted as having event rate coefficient c_v (the probability per unit time as time goes to 0). $\alpha_v^{(1)}$ individuals of species 1, ..., M interact according to event v, resulting in their removal from the system, and $\beta_v^{(1)}$ individuals of species 1, ..., M are introduced into the system. Thus, event v changes the populations by $\Delta_v = (\beta_v^{(1)} - \alpha_v^{(1)}, \cdots, \beta_v^{(M)} - \alpha_v^{(M)})$. Let $x_t = (x_t^{(1)}, \ldots, x_t^{(M)})$ be the populations of the species in the system at time t. A stochastic kinetic process initially in state x_0 at time $t = 0$ can be simulated through the Gillespie algorithm [39] by iteratively (1) sampling the time τ to the next event according to exponential distribution $\tau \sim \text{Exponential}(h_0(x_t, c))$, where $h_0(x, c) = \sum_{v=1}^{V} h_v(x_t, c_v)$ is the rate of all events and $h_v(x_t, c_v)$ is the rate of event v, (2) simulating event v according to categorical distribution $v \sim (\frac{h_1}{h_0}, \ldots, \frac{h_V}{h_0})$ conditional on event time τ, and (3) updating the system time $t \leftarrow t + \tau$ and populations $x \leftarrow x + \Delta_v$ until the termination condition is satisfied. In this algorithm, event rate $h_v(x_t, c_v)$ is the rate constant c_v multiplying a total of $\prod_{m=1}^{M}(x_t^{(m)})^{\alpha_v^{(m)}}$ different ways for individuals to interact in the system, assuming homogeneous populations. Exponential distribution is the maximum entropy distribution given the rate constant, and consequently it is most likely to occur in natural reactions [39].

A *partially observable discrete-event decision process* (PODEDP) is a partially observable Markov decision process [49] where the dynamics are defined by a discrete-event model. Let v_t be the *event* happening at time *t*. Let x_t be the *state* (populations of species or states of individuals), and y_t the *observation* about the state at time *t*. Let $b_t = p(x_t|y_{t,t-1,\cdots}, a_{t,t-1}) = b_t(b_{t-1}, y_t, a_{t-1})$ be the *belief state* — the probability distribution of the current system state estimated through observation-action history. Let a_t be the *control* (or *action*) variables influencing the event-rate constants at time t, $c(a_t) = (c_1(a_t), \cdots, c_V(a_t))$ where the

action or its distribution is determined by the belief state $a_t = \pi(b_t;\theta)$ or $\pi = p(a_t|b_t)$. Further let $p(y_t|x_t)$ be the *observation model*, $p(v_{t+1}, x_{t+1}|x_t, a_t)$ the *state transition model*, and $R(x_t, a_t)$ the *immediate reward* at time t. The PODEDP problem (Eq. 2) is to maximize the expected future reward of the discrete-time process defined by the probability measure $p(a_{0:t}, v_{1:t}, x_{0:t}, y_{1:t})$ (Eq. 3) by iteratively setting a_t from a belief state b_t that summarizes the observation-control history $(y_{1:t-1}, a_{0:t-1})$, where the indicator function $\delta(x_{t+1} \equiv x_t + \Delta_{v_{t+1}})$ is 1 if the current state is $x_{t+1} = x_t + \Delta_{v_{t+1}}$ and 0 otherwise, and $0 < \gamma < 1$ is a discount factor. The graphical model representations of a discrete-event process and a discrete-event decision process are given in Fig. 2.

$$\arg\max_{a_{0:\infty}} \mathbf{E}_{x_{0:\infty}, a_{0:\infty}, v_{0:\infty}, y_{0:\infty}} \left(\sum_{t=0}^{\infty} \gamma^t R(x_t, a_t) \right), \tag{2}$$

where

$$p\left(a_{0:T}, v_{1:T}, x_{0:T}, y_{1:T}\right) \tag{3}$$
$$= p(x_0) \prod_{t=0}^{T-1} p(a_t|b_t) p(v_{t+1}|x_t, a_t) \delta(x_{t+1} \equiv x_t + \Delta_{v_{t+1}}) p(y_{t+1}|x_{t+1}),$$

$$p(v_{t+1}|x_t, a_t; \theta) = \begin{cases} 1 - \sum_k \tau \cdot h_k(x_t, a_t), & v_{t+1} = \emptyset \\ \tau \cdot h_k(x_t, a_t), & v_{t+1} = k \end{cases}, \tag{4}$$

$$h_v(x, c_k) = c_v g_v(x) \text{ for } v = 1, \cdots, V, \text{ and } h_0(x, c) = \sum_{v=1}^{V} h_v(x, c_v). \tag{5}$$

A Markov decision process (MDP) is a framework for modeling decision making in situations where outcomes are partly random and partly under the control of a decision-maker. It is used in many fields, including robotics, manufacturing, optimal control, game theory, and economics. In recent years, it has seen increasing applications in intelligent transportation systems, including traffic-signal control, autonomous driving, and traffic assignment.

B. Modeling Traffic Dynamics with a Discrete-Event Model

In this subsection we define the traffic optimization problem in the framework of PODEDP (Eq. 2), specifying the utility, states, events, observations, and control variables. We consider the movement of thousands of vehicles in a city-scale transportation network with M locations and V events, for which the typical length of a control episode is one day discretized into 1440 steps of one minute each.

The *state* $x_t = (x_t^{(1)}, \cdots, x_t^{(M)}, t)$ is the number of vehicles at the M locations (road links and buildings) and the current time t. The *observation* $y_t = (y_t^{(1)}, \cdots, y_t^{(M)}, t)$ is the number of probe vehicles at the M locations and current time t, where the probe vehicles are randomly selected and constitute 10% of the total vehicle population. The *action* variables a_t are the probability of choosing a downstream road link after completing the current road link, and the event rate coefficient of leaving or entering buildings. These action variables change the event rate coefficients to make road usage more efficient. The *reward* function $R(x_t) = \sum_m \beta_{t,\text{perf}}^{(m)} x_t^{(m)} + \beta_{t,\text{trav}}^{(m)} x_t^{(m)}$ emulates the Charypar-Nagel scoring function [14] to reward performing the correct activities at locations and penalize traveling on roads, where $\beta_{t,\text{trav}}^{(m)}$ and $\beta_{t,\text{perf}}^{(m)}$ are the score coefficients.

Let x_{ttl} be the total number of vehicles in the system and y_{ttl} the total number of observed vehicles. The *observation model* of observing $y_t^{(m_j)}$ "probe" vehicles at location j conditioned on $x_t^{(m_j)}$ vehicles in total is $p(y_t^{(m_j)}|x_t^{(m_j)}) = C_{x_t^{(m_j)}}^{y_t^{(m_j)}} \cdot C_{x_{\text{ttl}} - x_t^{(m_j)}}^{y_{\text{ttl}} - y_t^{(m_j)}} / C_{x_{\text{ttl}}}^{y_{\text{ttl}}}$, where $C_a^b = a!/(b! \cdot (a-b)!)$ is a-choose-b.

We implement the *state transition model* $p(v_t, x_{t+1} \mid x_t, a_t)$ using discrete-event dynamics, where there is a single type of *event* $p \cdot m_1 \overset{c_{m_1 m_2}}{\rightarrow} p \cdot m_2$. A vehicle p moving from one location m_1 to the next m_2 with event rate coefficient $c_{m_1 m_2}$ changes the number of vehicles at the two locations to $x_t^{(m_1)} - 1$ and $x_t^{(m_2)} + 1$ respectively. The *event rate coefficients* $c_{m_1 m_2}$ (the probability of event per unit time as time goes to 0) of finishing the current road link is a function of the numbers of vehicles per lane per unit length, the maximum flow per lane, the speed limit, the length of the road link, and the road type (freeway, arterial road, or local road), implemented with a deep neural network and trained to best match the queue-network MATSim traffic dynamics [14]. In this way, we capture a variety of traffic behaviors involving traffic lights and traffic flow through a model-free approach.

Travel time modeled on a road link using an exponential random variable with matching average travel time is approximate. Nevertheless, such a model strikes a balance between capturing high-fidelity dynamics and being amiable to gradient-based machine learning algorithms. Because traffic-state estimation is driven by noisy observations, the inferred traffic state will be constrained by what the observations prescribe and thus will not stray far from the ground truth. This is different from pure simulation, where an approximate model can drive the system state to an unrealizable position. The optimal control from partial observations in this paper is a closed-loop control, such that any undesirable effect caused by model inaccuracy and randomness in the dynamics is corrected in the next time step. This approach is different from open-loop controls in classic transportation simulators for policy research.

IV. Tracking and Optimal Control

In this section, we describe a particle filter algorithm that tracks traffic state (Alg. 1, Sec. IV-A) and implements optimal control from partial observations (Alg. 2, Sec. IV-B).

A. Tracking with Particle Filter

In this subsection, we derive a particle filter algorithm to track vehicles counts (belief state) at different road links and buildings using the observed probe-vehicle counts at those locations in order to establish optimal control of transportation network dynamics. The usage of particle filtering permits to deal with noisy observations because it aggregates all information included in the past observations as a probability distribution of the current traffic state — numbers of vehicles on the links and locations. We also derive particle smoothing to back trace the evolution of particles.

Let y_t be a noisy observation of system state x_t at time t, and a_t be the control. A particle filter (sequential Monte

Carlo method) approximates the posterior probability distribution of a stochastic process through maintaining a collection of particles x_t^k and particle indices $i_t^k \in \{1, \cdots, K\}$ to represent the likely system state x_t, where $k = 1, \cdots, K$ and $t = 1, \cdots, T$. Inference with the particle filter involves tracking the evolution of a stochastic process by alternating between particle *prediction* and *updating*. In the *prediction* step, the particles at the next time step t are sampled according to the transition kernel $x_t^k \sim p(x_t \mid x_{t-1}^{i_{t-1}^k})$. In the *updating* step, the particle indices are resampled according to the observation likelihood $i_t^k \mid x_t^1, \ldots, x_t^K, y_t \sim \text{Categorical}\big(p(y_t \mid x_t^1), ..., p(y_t \mid x_t^K)\big)$. With the resulting particles and particle indices, we use the empirical probability $\frac{1}{K}\sum_k \delta_{x_t^k}(x_t)$ to approximate $p(x_t|y_1, \ldots, y_{t-1})$ and use $\frac{1}{K}\sum_k \delta_{x_t^{i_t^k}}(x_t)$ to approximate $p(x_t|y_1, \ldots, y_t)$, where δ is an indicator function.

To track a discrete event process initially at state x_0 at time $t_0 = 0$, we initialize particle positions and indices as $x_0^1, \cdots, x_0^K = x_0$ and $i_0^1 = 1, \cdots, i_0^K = K$, and alternate between the following prediction step and updating step.

In the *prediction* step, we sample particle positions $x_{t+1}^1 \sim p(x_{t+1}|x_t^{i_t^1}, a_t), \cdots, x_{t+1}^K \sim p(x_{t+1}|x_t^{i_t^K}, a_t)$ at time $t+1$ from the particles $x_t^{i_t^1}, \cdots, x_t^{i_t^K}$ at time t. Specifically, we sample event v_{t+1}^k according to how likely it is that different events will occur conditioned on system state $x_t^{i_t^k}$ for $k = 1, \cdots, K$ and action a_t (Eq. 6), and update $x_{t+1}^k = x_t^k + \Delta_{v_{t+1}^k}$ accordingly (Eq. 7). Because the resampled particles are distributed according to $x_t^{i_t^k} \sim p(x_t|y_{1:t}, a_{1:t-1})$, the sampled particles are distributed according to $x_{t+1}^k \sim p(x_{t+1}|y_{1:t}, a_{1:t})$.

The likelihood of particles x_{t+1}^k are $p(y_{t+1}|x_{t+1}^k)$ with respect to the observation y_{t+1}. To avoid particle degeneracy, we perform a *updating* step to eliminate particles with low likelihood and duplicate particles with high likelihood (Eq.8). After the particle-updating step, all particles are distributed according to $p(x_{t+1}|y_{1:t+1}, a_{1:t})$, and all have the same likelihood.

$$v_{t+1}^k|a_t, x_t^{i_t^k} \sim \text{Cat}(1 - \frac{h_0(x_t^{i_t^k}, a_t)}{\gamma}, \frac{h_1(x_t^{i_t^k}, a_t)}{\gamma}, \cdots, \frac{h_V(x_t^{i_t^k}, a_t)}{\gamma}), \tag{6}$$

$$x_{t+1}^k = x_t^{i_t^k} + \Delta_{v_{t+1}^k}, \tag{7}$$

$$i_{t+1}^k|x_{t+1}^{1:K}, y_{t+1} \sim \text{Cat}(p(y_{t+1}|x_{t+1}^1), \cdots, p(y_{t+1}|x_{t+1}^K)). \tag{8}$$

To derive a particle trajectory from the posterior distribution of a stochastic kinetic process with respect to observations, we trace back the events that lead to the particles $x_T^{i_T^k}$ for $k = 1, \cdots, N$:

$$x_0, a_0, v_1^{j_1^k}, x_1^{j_1^k}, a_1, \cdots, v_T^{j_T^k}, x_T^{j_T^k}, a_T, \tag{9}$$

where $j_T^k = i_T^k, j_{T-1}^k = i_{T-1}^{j_T^k}, j_{T-2}^k = i_{T-2}^{j_{T-1}^k}, \cdots, j_1^k = i_1^{j_2^k}$.

The particles $x_t^{i_t^1}, \cdots, x_t^{i_t^K}$ form an approximation of the forward probability $p(x_t|y_{1,\cdots,t})$ and likelihood $p(y_{1,\cdots,T})$. The ancestral lines of the particles $x_T^{i_T^k}$, where $k = 1, \cdots, K$, form an approximation of the posterior distribution of the stochastic process conditioned on observations, where δ is an indicator function:

$$\hat{p}(x_t|y_{1:t}) = \frac{1}{K}\sum_k \delta(x_t^{i_t^k} \equiv x_t) \overset{K\to\infty}{\longrightarrow} p(x_t|y_{1:t}), \tag{10}$$

$$\hat{p}(y_{1:T}) = \prod_t \hat{p}(y_t|y_{1:t-1}) = \prod_t \tfrac{1}{K}\sum_k p(y_t|x_t^k) \overset{K\to\infty}{\longrightarrow} p(y_{1:T}), \tag{11}$$

$$\hat{p}(x_{1:T}|y_{1:T}) = \tfrac{1}{K}\sum_k \delta((x_{1:T}^{j_{1:T}^k}) \equiv (x_{1:T})) \overset{K\to\infty}{\longrightarrow} p(x_{1:T}|y_{1:T}). \tag{12}$$

Overall, we develop a particle-based algorithm to update the belief state and calibrate the parameters of a discrete event decision process (Algorithm 1).

ALGORITHM 1: Particle filtering, smoothing, and parameter learning for discrete event decision process

Input: Observations $y_1, \cdots, y_T$ and control inputs $a_1, \cdots, a_T$ of a discrete event decision process (Eq. 3).

Output: Belief state $b_t \approx \hat{p}(x_t|y_{1:t})$ (Eq. 10) where particles $(v_t^{i_t^k}, x_t^{i_t^k})_{t=1:T}^{k=1:K}$ are sampled from particle filter, and particle trajectories $(v_t^{j_t^k}, x_t^{j_t^k})_{t=1:T}^{k=1:K}$ are sampled from particle smoother.

Procedure:

- Initialize $x_0^1 = \cdots = x_0^N = x_0$, $i_0^1 = 1, \cdots, i_0^K = K$.
- (Filtering) For $t = 1, \cdots, T$ and $k = 1, \cdots, K$, sample v_t^k and i_t^k according to Eq. 6, 7 and 8.
- (Smoothing) Back-track particle trajectory from $x_T^{i_T^k}$ according to Eq. 9, for $k = 1, \cdots, K$.

Calibration: Maximize log likelihood $\log \hat{p}(y_{1:T})$ (Eq. 11) with gradient ascent.

B. Optimal Control with Particle Filter

In this subsection, we derive a particle-based algorithm to identify the optimal control of a complex system from our estimation of the current system state (belief state), using the equivalence between the state-value function of a Markov decision process and the probability of receiving the reward from a mixture of finite-time Markov decision processes [50]. This equivalence enables the translation of the policy-evaluation and policy-improvement steps in a policy iteration algorithm into the expectation and maximization steps in an expectation maximization (EM) algorithm, and the application of a large variety of approximate inference algorithms for dynamic Bayesian networks to solve intractable optimal control problems. In particular, it is based on the following derivation:

$$\begin{aligned}\mathbf{E}\sum_{t=0}^{\infty}\gamma_t R(a_t, x_t) &= \sum_{t=0}^{\infty}\gamma_t \mathbf{E}\left(R(a_t, x_t)\right) \\ &\propto \sum_{T=0}^{\infty} p(T)\sum_{t=0}^{T} p(t)\mathbf{E}(p(R=1|a_t, x_t)),\end{aligned} \tag{13}$$

where $\gamma_t \propto \sum_{T=0}^{\infty} p(T)p(t)\delta_{t\le T}, p(R=1|a_t, x_t) \propto R(a_t, x_t)$.

Eq. 13 connects the expected future reward of a Markov decision process and the probability of receiving a binary

reward in a mixture of finite time Markov decision processes (T, t, ξ_t, R). This finite time Markov decision process executes the same plan as the original Markov decision process up to a terminal time t, it generates a state-action trajectory $\xi_t = x_0, a_0, \cdots, x_t, a_t$, and it receives a binary reward with probability $p(R = 1|x_t, a_t) \propto R(x_t, a_t)$. In Eq. 13, γ_t is a discount factor. Corresponding to the expected discounted cumulative future reward with $\gamma_t = \gamma^t$, we select $p(T) = (1-\delta)\delta^t$ and $p(t) = (1-\frac{\gamma}{\delta})(\frac{\gamma}{\delta})^t$. Corresponding to the expected finite-horizon future reward with $\gamma_t = \delta(t \leq H)$, we select $p(T) = \delta(H \equiv T)$ and $p(t) = 1/(H+1)$, where indicator function $\delta(T \equiv H) = 1$ when $T = H$ and 0 otherwise, and $\delta(t \leq H) = 1$ when $t \leq H$ and 0 otherwise.

To identify optimal control with the EM algorithm in a discrete event decision process, we maximize the expected log likelihood $\mathbf{E}_{T,t,\xi_t} \log p(T, t, \xi_t, R = 1; \theta)$ by alternately identifying the typical state-action sequences generated by a policy that leads to reward ($p(T, t, \xi_t|R = 1; \theta)$) in the expectation step (E-step) and tuning the control parameters of the policy (θ) so that these typical sequences lead to reward with higher probabilities in the maximization step (M-step). The EM algorithm is an iterative algorithm that searches for the parameters to maximize the expected log likelihood over the posterior probability distribution of the latent variables conditional on the observations. Here, the likelihood is proportional to the value function, the latent variables are a sequence of states and actions, and the observations are of whether a reward is received.

In E-step, we use importance sampling to approximate the proxy of future expected reward $p(R = 1)$ and the posterior probability $p(x_\tau, a_\tau \mid R = 1, \tau \leq t)$ induced in Eq. 13. Specifically, we sample $T^k \sim p(T)$ and $\xi^k \sim p(\xi_T|T^k)$ for $k = 1, \cdots, K$, we approximate the prior distribution $p(T, \xi_T)$ with sample distribution $\hat{p}(T, \xi_T) = \frac{1}{K}\sum_{k=1}^{K} \delta(T^k, \xi^k) \equiv (T, \xi_T))$, and use importance weight $\sum_{t=0}^{T} p(t)p(R = 1|x_t^k, a_t^k)$ to approximate the posterior distribution, where δ is an indicator function.

$$T^k, a_{0:T^k}^k, x_{0:T^k}^k, v_{1:T^k}^k \sim \tag{14}$$
$$P(T^k)b(x_0)\prod_{t=0}^{T^k} p(a_t|x_t;\theta)p(v_t|a_t,x_t)\delta(x_{t+1} \equiv x_t + \Delta_{v_{t+1}})$$

$$\mathbf{E}\sum_{t=0}^{\infty}\gamma_t R(a_t,x_t) \propto p(R=1) \approx \frac{1}{K}\sum_{k,t} p(t)p(R=1|x_t^k,a_t^k) \tag{15}$$

$$\hat{p}(x_\tau = x, a_\tau = a|R = 1, \tau \leq t) \tag{16}$$
$$= \frac{\sum_{k=1}^{K}\delta(x \equiv x_\tau^k)\delta(a \equiv a_\tau^k)\sum_{t=\tau}^{T} p(t)p(R = 1|x_t^k, a_t^k)}{\sum_{k=1}^{K}\sum_{t=\tau}^{T} p(t)p(R = 1|x_t^k, a_t^k)}.$$

The posterior probability (Eq. 16) is the fraction of expected future discounted reward received from $x_\tau^k = x, a_\tau^k = a$ over the total expected future discounted reward received after τ, averaged over sample paths $\{(T^k, \xi^k) : k\}$.

In M-step, we iteratively maximize the expected log likelihood of receiving a reward. The optimal control θ is consequently set such that actions a appears in proportion to the future rewards.

$$\mathbf{E}_{\theta^{\text{old}}} \log p(\xi_T, T, R = 1; \theta) = \cdots + \mathbf{E}_{\pi^{\text{old}}} \log p(a_t, x_t|R = 1; \theta),$$
$$\Rightarrow p(a|x;\theta) = \frac{\sum_{k,\tau}\delta(x_\tau \equiv x)\delta(a_\tau \equiv a)\sum_{t=\tau}^{T^k} p(t)p(R=1|x_t^k,a_t^k)}{\sum_{k,\tau}\delta(x_\tau \equiv x)\cdot\sum_{t=\tau}^{T^k} p(t)p(R=1|x_t^k,a_t^k)}. \tag{17}$$

To summarize, we develop an algorithm 2 to control a complex system from a discrete event model and noisy observations.

ALGORITHM 2: Optimal control from belief state with Particle Filter

Input: Belief state (Eq. 10) of discrete event process (Eq. 3) at time 0 as particles, $\{x_0^k : k = 1{:}K\}$. Initial policy θ at time 0.
Output: Optimal action a_t according to Eq. 17.
Calibration: Improve policy θ through policy iteration.
- E step. For $k = 1{:}K$: sample T^k, ξ^k according to Eq. 14
- M step. Upate θ according to Eq. 17.

V. Tracking and Planning in City-Scale Transportation Networks

In this section, we benchmark our framework with other state-of-the-art algorithms on tracking and optimizing the travel plans in a city-scale transportation network from noisy observations of network dynamics.

A. Data Description

We evaluate the performance of our framework on five datasets of human mobility: (1) SynthTown, (2) Berlin, (3) Santiago de Chile, (4) Dakar, and (5) NYC Taxicab. We use as varied data as possible to demonstrate that our proposed framework is useful in more than a narrow range of cases and is fair. The different cases indeed show different levels of uncertainty in traffic state estimation, different travel times, and different on-time arrival rates. Nevertheless, the proposed traffic tracking and control algorithms outperforms the state of the art.

The SynthTown dataset is comprised of a synthesized network of one home location, one work location, and 23 single-direction road links (Fig. 3) to characterize the trips of 2000 synthesized inhabitants going to work in the morning and returning home in the afternoon [14]. The prediction problem is to estimate the vehicle counts at home, at work, and at links 1-23 in the present time, 10 minutes later, and 60 minutes later from observations of the 200 "probe" inhabitants collected at link 1 and link 20. The control problem is to maximize a proxy of the Charypar-Nagel scoring function [14] from setting control variables according to these observations. We use this dataset to show the details of tracking and control results.

The Berlin dataset is comprised of a network of 11 thousand nodes and 24 thousand single-direction car-only links derived from OpenStreetMap; and the trips of 9 thousand synthesized vehicles representing the travel behaviors of three million inhabitants [20]. To make the problem small enough that algorithms with bigger time-complexity can run and have performances compared with our algorithm, we aggregate the

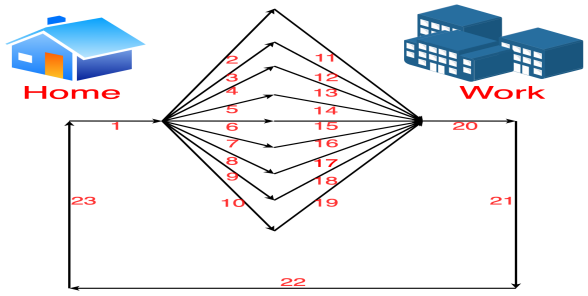


Fig. 3. SynthTown road network, which contains 2 facilities, Home and Work, and 23 road links labeled from 1 to 23.

24 thousand road links into 1539 clusters with a walk-trap algorithm [51]. The synthesized daily trips have been validated based on extensive, regularly-conducted travel surveys and constitute a quality representation of road transport demand. This data set is the result of a generalizable approach to synthesize individual-level behaviorally-sound trip diaries from easily accessible input data, since collecting the trip diaries of real-world people is plagued with privacy issues.

The Santiago de Chile dataset is comprised of a network of 23 thousand nodes and 38 thousand single-direction car-only links derived from OpenStreetMap; and the trips of 665 thousand synthesized vehicles representing the travel behaviors of six million inhabitants in car, walking and public transportation modals [21]. The daily trips in the Santiago de Chile dataset were initialized from cloning the sequences of activities (starting time and duration of home, work, school, shopping, leisure, visit and health) and travel mode of 60 thousand individuals (from 18 thousand households) from publicly-accessible travel diaries, and modified through physical simulation and a co-evolutionary algorithm (with MATSim) to maximize the overall utility of the system. The resulting daily trips are compatible with travel modals' distributions and observed traffic counts at count stations. This data set represents the case where we can get travel diaries with fine temporal and spatial resolution for a significant and representative fraction of a population from publicly-accessible travel diaries.

The Dakar dataset is comprised of a network of 8 thousand single-direction road links derived from OpenStreetMap and 12 thousand real-world vehicle trips derived from the "Data for Development (D4D)" data sets based on the Call Detail Records (CDR) of over 9 million Sonatel customers in Senegal (out of 15 million total population) through year 2013 [22]. From data set 2, we identify the home and work/school locations of each user as randomly picked locations from the most appeared sites during 7am - 7pm and 7pm - 7am respectively. Then, we sample an activity-trip sequence for each user to match her/his sequence of mobility records (in data set 2) from a Markov chain model describing how s/he performed various activities (home, work, school, shopping, etc). This data set represents the case where we can get travel diaries with fine temporal and spatial resolution for a significant and representative fraction of a population through mobile phones.

The NYC TaxiCab dataset[1] is comprised of a network of 7 thousand nodes and 11 thousand single-direction road links derived from OpenStreetMap and an average of 1 million daily trips of taxicabs and for-hire vehicles (including Uber, Lyft, Via and Juno) throughout 2018. Each trip record contains pick-up and drop-off zones among the 236 zones in New York City, and pick-up and drop-off data and time, among other information. The trip records are made publicly accessible by the New York City Taxi and Limousine Commission (an agency responsible for licensing and regulating New York City's taxi cabs, for-hire vehicles, commuter vans, and para-transit vehicles). Together with many other open data sets through the City's Open Data portal[2], TLC's trip data has a big impact in making the city street smart. Here, we use the data to predict the behavior of all taxicabs and for-hire vehicles from observing a small fraction of them.

B. Tracking Transportation Dynamics

Benchmark algorithms: We firstly benchmark our framework — stochastic kinetic model with particle filter (PFSKM) — against a Deep Neural Network (DNN) [6], [7], a Recurrent Neural Network (RNN) [10], and an extended Kalman filter (EKF) [26], [27] in the task of continuously tracking the current and future traffic conditions. DNN represents the power of a general-purpose non-parametric model. We build a five layer Deep Neural Network (DNN): (i) an input layer accepting the observation history of probe vehicles at selected locations, (ii) three hidden layers, and (iii) one output layer generating the inferred distribution of all vehicles at all locations. RNN exploits the temporal structure that recursively takes the inferred result from the previous cell as well as the current observations as input, and output the estimated vehicle distribution. Both DNN and RNN are trained with 30 days of synthesized mobility data from MATSim until obtaining optimum performance. The EKF, instead, assumes a Gaussian distribution between the time-indexed latent states, and we implement a standard EKF procedure that alternates between predicting and updating steps. We trained the DNN and RNN models with stochastic gradient descent, and trained the EKF model with expectation maximization.

Evaluation metric: We use two metrics to evaluate the performance of our model: coefficient of determination (R^2) and mean squared error (MSE). We use R^2 to evaluate the goodness of fit between a time series of the estimated vehicle counts at a location and the ground truth. Let f_t be the estimated vehicle count at time t, y_t the ground truth and $\bar{y}$ the average of y_t. We define $R^2 = 1 - \sum_t (f_t - y_t)^2 / \sum_t (y_t - \bar{y}_t)^2$. A higher R^2 indicates a better fit between the estimated time series and the ground truth, with $R^2 = 1$ indicating a perfect fit and $R^2 < 0$ a fit worse than using the average. We use MSE to measure the average squared error difference between the estimated vehicle counts at all locations at a time t and the ground truth. A lower MSE represents a more precise prediction. Let $f^{(i)}$ be the estimated vehicle

[1] http://nyc.gov/tlcopendata
[2] https://opendata.cityofnewyork.us/

(a) Ground truth (b) Particle filter

Fig. 4. Comparing estimated and ground truth traffic densities through two snapshots taken at the same time (black dots: vehicles in the ground truth; red dots: vehicles from a particle trajectory; green dots: probe vehicles).

count at location i and $y^{(i)}$ the ground truth. We define $\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y^{(i)} - f^{(i)})^2$.

Results' visualization: One benefit of using a discrete-event model with particle filter is that we can qualitatively visualize how vehicles move in a city-scale road network in accordance with "probe" vehicle locations, and how traffic policies change vehicle behavior. Fig. 4 compares the distributions of vehicles in the ground truth and in a particle trajectory (Eq. 9) using as an example the Dakar data set through two snapshots taken at the same time. It can be observed from the figures that the vehicle density in ground truth and estimation agree with each other, and both are proportional to the density of "probe" vehicles.

Evaluation results: Figure 5 summarizes the MSE and R^2 performance statistics of the four models for a vehicle-tracking task — estimating the numbers of vehicles up to now, with short-term (10 minutes) and long-term prediction (1 hour) on all the datasets. The Dakar dataset is simply too large for DNN, RNN, and EKF, which demonstrates the superior scalability of PFSKM. PFSKM has the lowest MSE across different times of day, followed in order by DNN, EKF, and RNN (top row, lower is better). PFSKM also has the highest R^2 across different locations, followed by DNN, EKF, and RNN (bottom row, higher is better). PFSKM outperforms RNN and DNN because it can explicitly leverage problem-specific structures such as road topology. PFSKM outperforms EKF because it can work with arbitrary probability distributions, and sometimes a Gaussian assumption is not a good approximation for real-world applications. This comparison also points to new developments in neural network architectures that are either regularized by event-based structures of a complex system or can learn such structures explicitly.

Comparing detailed predictions using SynthTown data: Fig. 6 shows how PFSKM, DNN, RNN, and EKF predict the number of vehicles at different locations of SynthTown one hour ahead of time throughout the day from observations of probe vehicles (10% of the total) at links 1 and 20 only. The x-axis indicates the hour of the day, the y-axis shows the number of vehicles at different locations (home, work, and road segments marked on the left), and the ground truth (GT) serves as the frame of reference.

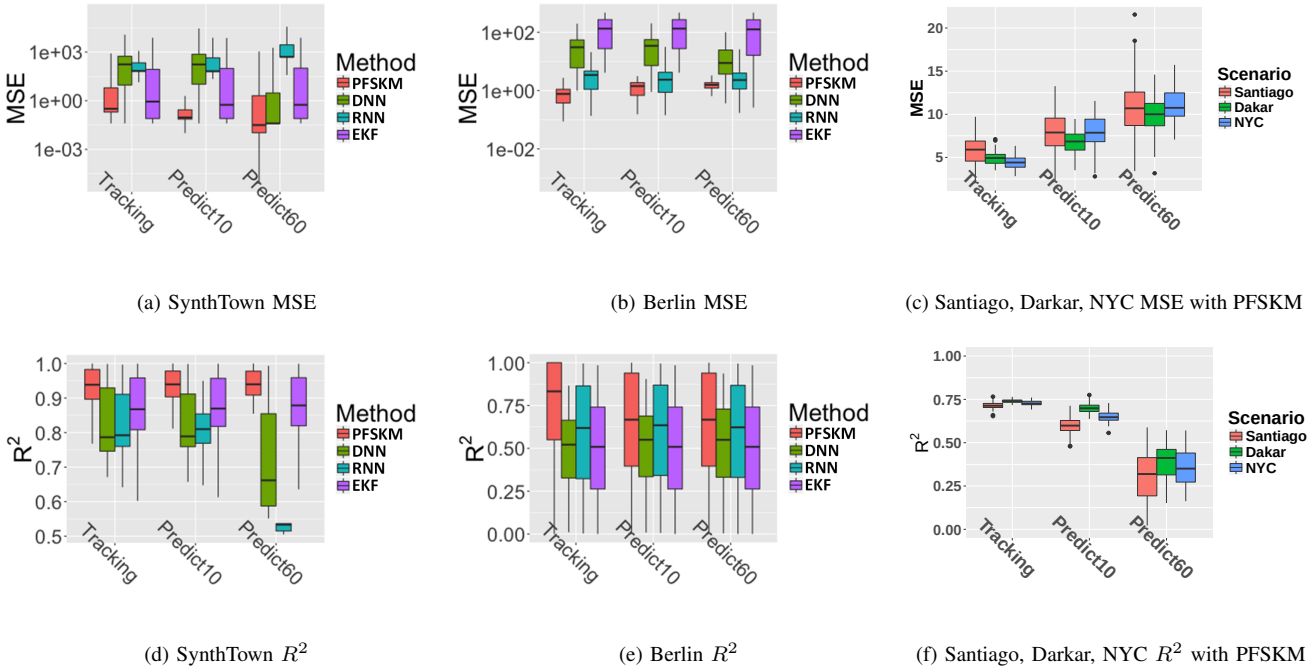
All four algorithms perform well, indicating that they all learn the structure in the dynamics. In fact, there is little uncertainty about the traffic dynamics at SynthTown if the number of vehicles on links 1 and 20 can be monitored, although with noise. RNN underperforms the other three algorithms because learning the structure of a dynamic system requires a huge training dataset. PFSKM estimation agrees with GT, and is closer than DNN and RNN estimations. This is because PFSKM explicitly leverages problem-specific structures such as road topology, while DNN and RNN must learn them implicitly and gradually. PFSKM is more accurate than EKF estimation because PFSKM can work with arbitrary probability distributions while EKF assumes Gaussianity. EKF and DNN agree well with GT at busy locations (home and work) but less well at locations with few people, which demonstrates that PFSKM better adopts dynamic changes.

C. Optimal Control in Transportation Dynamics

Benchmark algorithms: In the previous section we have demonstrated the tracking capability of our framework with particle filter. Now, we evaluate our framework for optimizing traffic against (i) a within-day re-planning baseline algorithm [52], (ii) a co-evolutionary algorithm implementing open-loop control [14], and (iii) an advantage actor-critic algorithm [53] implementing closed-loop control. The baseline algorithm (Baseline) optimizes agents' expected future rewards by considering the current traffic situation but not the plans of other agents. The co-evolutionary algorithm (CoEA) is the state-of-the-art algorithm for generating the equilibrium of daily activities and trips in transportation theory [14]. In CoEA, agents independently explore and exploit their plans through a genetic operator, then jointly execute and evaluate their plans in a simulator, and finally repeat this process until an equilibrium is reached [54]. The advantage actor-critic algorithm (A2C) uses a "critic" to estimate the traffic situation in terms of an action-value function and an "actor" which suggests optimal departure times and route choices in the direction suggested by the critic. We further use advantage to lower the variance.

Evaluation metric: We use three metrics to evaluate the different planning algorithms. The first is *average trip time* in minutes of all vehicles driving from home to work: a lower average trip time means better traffic. The second is *on-time arrival ratio*, which measures the percentage of people arriving to work on time. Finally, we use *expected reward* per vehicle per hour, where higher expected rewards mean better individual plans and a more efficient transportation network.

Comparing detailed behaviors on SynthTown data: Figure 7 shows the vehicle counts at the different locations of SynthTown throughout the day after executing different planning algorithms from the observations of probe vehicles (10% of the total) at links 1 and 20 only. The x-axis indicates the hours of the day, the y-axis shows the numbers of vehicles at different locations (home, work, and road segments marked on the left), and the baseline (Baseline) serves as the frame of reference. Note that vehicles applying the policy from our framework best satisfy the requirements of all individuals. Indeed, at 9am our framework has the highest number of people arriving at work on time, followed by within-day re-planning Baseline, A2C, and CoEAs. Similarly, at 5pm our framework has the highest number of people arriving back at home, while under the other three policies most are either still at work or slowed by congestion on roads. Finally, analyzing



(a) SynthTown MSE (b) Berlin MSE (c) Santiago, Darkar, NYC MSE with PFSKM

(d) SynthTown R^2 (e) Berlin R^2 (f) Santiago, Darkar, NYC R^2 with PFSKM

Fig. 5. Performance of PFSKM, DNN, RNN and EKF on SynthTown, Berlin, Dakar, Santiago de Chile and NYC Taxicab datasets using MSE (top, lower MSE indicates better performance) and R^2 (bottom, higher R^2 indicates better performance).

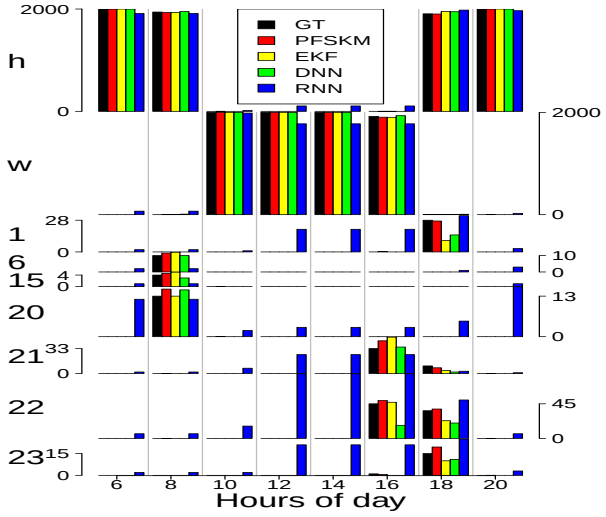


Fig. 6. Predicted vehicle counts at home, work, and different road segments of SynthTown (y-axis, with locations marked on the left) at different hours of a day (x-axis) from the observations of probe vehicles (10% of the total) at links 1 and 20 only. Algorithms with a good performance should lead to a prediction closer to the ground truth (GT).

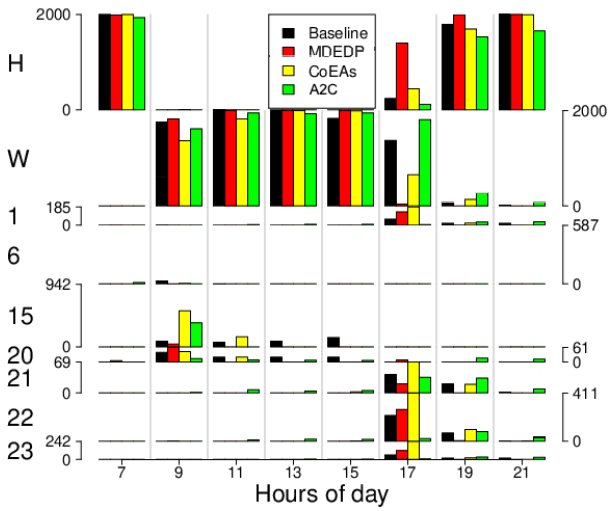


Fig. 7. Vehicle counts at home, work, and various road segments of SynthTown (y-axis, with locations marked on the left) at different hours of a day (x-axis) after executing different planning and control algorithms from the observations of probe vehicles (10% of the total) at links 1 and 20 only. Algorithms with a good performance should lead to lower trip time and higher on-time arrival ratio.

the figure horizontally, the people in our framework spend the least time on roads (links 1, 6, 15, 20, 21, 22, and 23), and the most time doing useful activities at locations (home and work).

Comparing summary performance metrics: Table I compares average trip time, on-time arrival ratio, and average unit reward statistics of the four models using the SynthTown, Berlin, Dakar, and Santiago de Chile datasets. The Berlin, Dakar, and Santiago de Chile datasets are too large for the A2C model to run, demonstrating the superior scalability of our framework and CoEA. This comparison leads us to the same conclusion as the detailed comparison on the SynthTown data. Specifically, our framework has the lowest average trip time, the highest on-time arrival ratio, and highest expected reward among all datasets. The within-day re-planning algorithm works better than the co-evolutionary algorithm because the former uses instantaneous traffic information for trip planning in terms of average trip time, on-time arrival ratio, and the Charypar-Nagel score. Our algorithm based on solving a partially observable Markov decision process problem works better than the within-day re-planning algorithm, because we specifically optimize the Charypar-Nagel scoring function

TABLE I
COMPARING OUR FRAMEWORK, COEA, AND A2C FOR *average trip time* IN MINUTES, *on-time arrival ratio*, AND *expected reward* PER VEHICLE PER HOUR. RESULTS ARE OBTAINED USING THE SYNTHTOWN, BERLIN, DAKAR, AND SANTIAGO DE CHILE DATASETS.

Dataset	Models	Average trip time	On-time arriving ratio	Expected reward
SynthTown	Baseline	45.57	0.88	1.05
	Our framework	**31.49**	**0.89**	**2.93**
	CoEA	55.47	0.85	-0.05
	A2C	50.64	0.88	0.30
Berlin	Baseline	39.65	0.72	-20.65
	Our framework	**38.38**	**0.86**	**-4.83**
	CoEA	40.27	0.68	-54.00
Dakar	Baseline	29.13	0.86	-5.27
	Our framework	**28.12**	**0.90**	**0.78**
	CoEA	30.30	0.85	-10.06
Santiago	Baseline	36.84	0.81	-15.33
	Our framework	**35.36**	**0.83**	**-7.58**
	CoEA	38.30	0.75	-20.79

through traffic prediction and optimization within POMDP.

VI. CONCLUSIONS AND DISCUSSIONS

The capability to sense the movement of millions of vehicles through mobile phones and to offer drivers trip plans has enabled a new way of controlling city traffic dynamics by turning transportation big data into insights and actions in a closed-loop. In this paper, we have presented a study showing that floating car data can indeed be utilized to offer optimal departure times and route choices associated with lower average trip time, higher on-time arrival ratio, and higher Charypar-Nagel score compared with how people normally travel, validated both in an agent-based transportation simulator and in the real world. The study is based on optimizing a partially observable discrete-time decision process and is evaluated in one synthesized scenario, one partly synthesized scenario, and three real-world scenarios.

The results show that the framework of turning transportation big data into insights and actions in the real world is very much worth further research. For example, while the study in this paper is based on maximizing the Charypar-Nagel scoring function, further research is needed to identify the utilities of people in the real world in the framework of inverse reinforcement learning [23], [55]. While a particle filter seems to be sufficient for traffic-prediction and city traffic optimization in this study, further research is needed to lower the variance of Monte Carlo integration [56], [57]. In this paper we approximated the queuing-network traffic dynamics of MATSim with a discrete-event process, but further work to turn transportation simulators into a reinforcement learning environment [58] will be useful to facilitate this "living lab" style of research.

REFERENCES

[1] S. An, J. Cui, and L. Li, "Agent-based approach to model commuter behaviour's day-to-day dynamics under pre-trip information," *IET Intelligent Transport Systems*, vol. 5, no. 1, pp. 70–79, 2011.

[2] K. Kung, K. Greco, S. Sobolevsky, and C. Ratti, "Exploring universal patterns in human home-work commuting from mobile phone data," *PloS One*, vol. 9, no. 6, p. e96180, 2014.

[3] Y. Zheng, Y. Liu, J. Yuan, and X. Xie, "Urban computing with taxicabs," in *Proceedings of the 13th International Conference on Ubiquitous Computing (Ubicomp)*, 2017, pp. 89–98.

[4] E. J. Horvitz, J. Apacible, R. Sarin, and L. Liao, "Prediction, expectation, and surprise: Methods, designs, and study of a deployed traffic forecasting service," in *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence (UAI2005)*, 2005, pp. 275–283.

[5] F. Yang, B. Liu, and W. Dong, "Optimal control of complex systems through variational inference with a discrete event decision process," in *Proceedings of the 2019 International Conference on Autonomous Agents & Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

[6] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach." *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2015.

[7] N. G. Polson and V. O. Sokolov, "Deep learning for short-term traffic flow prediction," *Transportation Research Part C: Emerging Technologies*, vol. 79, pp. 1–17, 2017.

[8] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, "Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction," *Sensors*, vol. 17, no. 4, p. 818, 2017.

[9] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction." in *AAAI*, 2017, pp. 1655–1661.

[10] Z. Zhao, W. Chen, X. Wu, P. C. Chen, and J. Liu, "Lstm network: a deep learning approach for short-term traffic forecast," *IET Intelligent Transport Systems*, vol. 11, no. 2, pp. 68–75, 2017.

[11] A. K. Ziliaskopoulos, "A linear programming model for the single destination system optimum dynamic traffic assignment problem," *Transportation science*, vol. 34, no. 1, pp. 37–49, 2000.

[12] M. Zolfpour-Arokhlo, A. Selamat, S. Z. M. Hashim, and H. Afkhami, "Modeling of route planning system based on q value-based dynamic programming with multi-agent reinforcement learning algorithms," *Engineering Applications of Artificial Intelligence*, vol. 29, pp. 163–177, 2014.

[13] L. Smith, R. Beckman, and K. Baggerly, "Transims: Transportation analysis and simulation system," Los Alamos National Lab., NM (United States), Tech. Rep., 1995.

[14] A. Horni, K. Nagel, and K. W. Axhausen, "The multi-agent transport simulation matsim," *Ubiquity, London*, vol. 9, 2016.

[15] S. Timotheou, C. G. Panayiotou, and M. M. Polycarpou, "Distributed traffic signal control using the cell transmission model via the alternating direction method of multipliers," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 919–933, 2015.

[16] S. Lin, B. De Schutter, Y. Xi, and H. Hellendoorn, "Efficient network-wide model-based predictive control for urban traffic networks," *Transportation Research Part C: Emerging Technologies*, vol. 24, pp. 122–140, 2012.

[17] W. Genders and S. Razavi, "Using a deep reinforcement learning agent for traffic signal control," *arXiv preprint arXiv:1611.01142*, 2016.

[18] N. Casas, "Deep deterministic policy gradient for urban traffic light control," *arXiv preprint arXiv:1703.09035*, 2017.

[19] S. El-Tantawy, B. Abdulhai, and H. Abdelgawad, "Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (marlin-atsc): methodology and large-scale application on downtown toronto," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 3, pp. 1140–1150, 2013.

[20] D. Ziemke, K. Nagel, and C. Bhat, "Integrating cemdap and matsim to increase the transferability of transport demand models," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2493, pp. 117–125, 2015.

[21] B. Kickhöfer, D. Hosse, K. Turner, and A. Tirachini, "Creating an open matsim scenario from open data: The case of santiago de chile," *http://www. vsp. tuberline. de/publication: TU Berlin, Transport System Planning and Transport Telematics*, 2016.

[22] Y.-A. de Montjoye, Z. Smoreda, R. Trinquart, C. Ziemlicki, and V. D. Blondel, "D4d-senegal: the second mobile phone data for development challenge," *arXiv preprint arXiv:1407.4885*, 2014.

[23] F. Yang, A. Vereshchaka, Y. Zhou, C. Chen, and W. Dong, "Variational adversarial kernel learned imitation learning." in *AAAI*, 2020, pp. 6599–6606.

[24] F. Yang, B. Lepri, and W. Dong, "Optimal control in partially observable complex social systems," in *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, 2020, pp. 1557–1565.

[25] F. Yang, A. Vereshchaka, and W. Dong, "Predicting and optimizing city-scale road traffic dynamics using trajectories of individual vehicles," in

2018 IEEE International Conference on Big Data (Big Data). IEEE, 2018, pp. 173–180.

[26] Y. Wang, M. Papageorgiou, and A. Messmer, "Renaissance–a unified macroscopic model-based approach to real-time freeway network traffic surveillance," *Transportation Research Part C: Emerging Technologies*, vol. 14, no. 3, pp. 190–212, 2006.

[27] C. P. van Hinsbergen, T. Schreiter, F. S. Zuurbier, J. Van Lint, and H. J. Van Zuylen, "Localized extended kalman filter for scalable real-time traffic state estimation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 1, pp. 385–394, 2012.

[28] J. Van Lint and S. P. Hoogendoorn, "A robust and efficient method for fusing heterogeneous data from traffic sensors on freeways," *Computer-Aided Civil and Infrastructure Engineering*, vol. 25, no. 8, pp. 596–612, 2010.

[29] X. Xie, H. van Lint, and A. Verbraeck, "A generic data assimilation framework for vehicle trajectory reconstruction on signalized urban arterials using particle filters," *Transportation Research Part C: Emerging Technologies*, vol. 92, pp. 364–391, 2018.

[30] F. Calabrese, M. Colonna, P. Lovisolo, D. Parata, and C. Ratti, "Real-time urban monitoring using cell phones: A case study in rome," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 1, pp. 141–151, 2011.

[31] J. Ferreira Junior, E. Carvalho, B. Ferreira, C. de Souza, Y. Suhara, A. Pentland, and G. Pessin, "Driver behavior profiling: An investigation with different smartphone sensors and machine learning," *PloS One*, vol. 12, no. 4, p. e0174959, 2017.

[32] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional neural network: A deep learning framework for traffic forecasting," *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)*, 2018.

[33] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Graph convolutional recurrent neural network: Data-driven traffic forecasting," *The Sixth International Conference on Learning Representations (ICLR)*, 2018.

[34] Z. Zhu, C. Xiong, X. Chen, X. He, and L. Zhang, "Integrating mesoscopic dynamic traffic assignment with agent-based travel behavior models for cumulative land development impact analysis," *Transportation Research Part C: Emerging Technologies*, vol. 93, pp. 446–462, 2018.

[35] M. E. Ben-Akiva, S. Gao, Z. Wei, and Y. Wen, "A dynamic traffic assignment model for highly congested urban networks," *Transportation research part C: emerging technologies*, vol. 24, pp. 62–82, 2012.

[36] H. Rong, X. Zhou, C. Yang, Z. Shafiq, and A. Liu, "The rich and the poor: A markov decision process approach to optimizing taxi driver revenue efficiency," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, 2016, pp. 2329–2334.

[37] X. Yu, S. Gao, X. Hu, and H. Park, "A markov decision process approach to vacant taxi routing with e-hailing," *Transportation Research Part B: Methodological*, vol. 121, pp. 114–134, 2019.

[38] V. D. Blondel, A. Decuyper, and G. Krings, "A survey of results on mobile phone datasets analysis," *arXiv preprint arXiv:1502.03406*, 2015.

[39] D. T. Gillespie, "Stochastic simulation of chemical kinetics," *Annual Review of Physical Chemistry*, vol. 58, pp. 35–55, 2007.

[40] D. J. Wilkinson, *Stochastic modelling for systems biology*. CRC press, 2011.

[41] P. J. Goss and J. Peccoud, "Quantitative modeling of stochastic systems in molecular biology by using stochastic petri nets," *Proceedings of the National Academy of Sciences*, vol. 95, no. 12, pp. 6750–6755, 1998.

[42] J. W. Forrester, *Urban dynamics*. MIT Press, Cambridge, 1969, vol. 114.

[43] A. Borshchev, *The Big Book of Simulation Modeling: Multimethod Modeling with AnyLogic 6*. AnyLogic North America, 2013.

[44] M. Opper and G. Sanguinetti, "Variational inference for markov jump processes," in *Advances in Neural Information Processing Systems*, 2008, pp. 1105–1112.

[45] V. Rao and Y. W. Teh, "Fast mcmc sampling for markov jump processes and extensions." *Journal of Machine Learning Research*, vol. 14, no. 1, pp. 3295–3320, 2013.

[46] U. Nodelman, C. R. Shelton, and D. Koller, "Continuous time bayesian networks," in *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*. Morgan Kaufmann Publishers Inc., 2002, pp. 378–387.

[47] A. Newell and H. A. Simon, *Human problem solving*. Prentice-Hall Englewood Cliffs, NJ, 1972, vol. 104.

[48] M. A. Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis, *Modelling with generalized stochastic Petri nets*. John Wiley & Sons, Inc., 1994.

[49] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[50] M. Toussaint, S. Harmeling, and A. Storkey, "Probabilistic inference for solving (po) mdps," Institute for Adaptive and Neural Computation, Tech. Rep., 2006.

[51] P. Pons and M. Latapy, "Computing communities in large networks using random walks." *J. Graph Algorithms Appl.*, vol. 10, no. 2, pp. 191–218, 2006.

[52] J. Illenberger, G. Flotterod, and K. Nagel, "Enhancing matsim with capabilities of within-day re-planning," in *2007 IEEE Intelligent Transportation Systems Conference*. IEEE, 2007, pp. 94–99.

[53] V. R. Konda and J. N. Tsitsiklis, "Actor-critic algorithms," in *Advances in neural information processing systems*, 2000, pp. 1008–1014.

[54] E. Popovici, A. Bucci, R. P. Wiegand, and E. D. De Jong, "Coevolutionary principles," in *Handbook of natural computing*. Springer, 2012, pp. 987–1033.

[55] F. Yang, A. Vereshchaka, C. Chen, and W. Dong, "Bayesian multi-type mean field multi-agent imitation learning," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[56] Z. Xu, W. Dong, and S. N. Srihari, "Using social dynamics to make individual predictions: variational inference with a stochastic kinetic model," in *Advances in Neural Information Processing Systems*, 2016, pp. 2783–2791.

[57] L. Fang, F. Yang, W. Dong, T. Guan, and C. Qiao, "Expectation propagation with stochastic kinetic model in complex interaction systems," in *Advances in neural information processing systems*, 2017, pp. 2029–2039.

[58] L. Khaidem, M. Luca, F. Yang, A. Anand, B. Lepri, and W. Dong, "Optimizing transportation dynamics at a city-scale using a reinforcement learning framework," *IEEE Access*, vol. 8, pp. 171 528–171 541, 2020.

Fan Yang is a PhD candidate in the Department of Computer Science and Engineering at the State University of New Your at Buffalo, USA. His research interests includes reinforcement learning, imitation learning, modeling, generative models, and probabilistic graphical models.

Alina Vereshchaka is a PhD candidate in the Department of Computer Science and Engineering at the State University of New Your at Buffalo, USA. Her current research interests include deep reinforcement learning, optimization and multi-agent modeling in stochastic environments. She has conducted studies in the application areas of optimization, transportation and healthcare.

Bruno Lepri received the Ph.D. degree in computer science from the University of Trento, Italy, in 2009. From 2010 to 2013, he was a joint Post-Doctoral Fellow at MIT Media Lab, Boston, MA, USA, and at Fondazione Bruno Kessler (FBK), Trento, Italy, where he is currently leading the Mobile and Social Computing Lab (MobS). He is also the Head of the Research of Data-Pop Alliance, the first thinktank on Big Data and Development co-created by Harvard Humanitarian Initiative, MIT Media Lab, and Overseas Development Institute.

Wen Dong is an Assistant Professor of Computer Science and Engineering with a joint appointment at the Institute of Sustainable Transportation and Logics at the State University of New York at Buffalo. His research focuses on developing machine learning and signal processing tools to study the dynamics of large social systems in vivo. He has a PhD degree from the M.I.T. Media Laboratory.